

⑩ 日本国特許庁 (J P)

⑪ 特許出願公開

⑫ 公開特許公報 (A) 昭60-118935

⑬ Int. Cl. 4

G 06 F 9/44  
11/00

識別記号

庁内整理番号

7361-5B  
7368-5B

⑭ 公開 昭和60年(1985)6月26日

審査請求 未請求 発明の数 1 (全4頁)

⑮ 発明の名称 コンパイラ障害自動検出方式

⑯ 特 願 昭58-226321

⑰ 出 願 昭58(1983)11月30日

⑱ 発 明 者 池 田 敬 昭 川崎市中原区上小田中1015番地 富士通株式会社内  
⑲ 出 願 人 富 士 通 株 式 会 社 川崎市中原区上小田中1015番地  
⑳ 代 理 人 弁 理 士 長 谷 川 文 廣 外 1 名

明 細 書

1. 発明の名称 コンパイラ障害自動検出方式

2. 特許請求の範囲

ソースプログラムを複数回のパスによりコンパイル処理するコンパイラを有するデータ処理装置において、上記コンパイラを用いたコンパイル処理の第1回のパスに要する処理時間と、第2回以降の各回ごとのパスに要する処理時間との間の比率について予測できる最大値を超えた所定の値が予め設定されているテーブル手段と、実際のソースプログラムについてコンパイル処理時に各回ごとのパスに要する処理時間を計測する手段と、第2回以降のパスごとに検した処理時間と第1回のパスに要した処理時間に上記テーブル手段に設定されている当該回の比率とを乗じた時間とを比較し、前者が後者よりも大きくなつたとき無限ループ状態と判定する手段とを有し、該判定手段が無限ループ状態と判定したパスにおいて異常終了

処理を行なうことを特徴とするコンパイラ障害自動検出方式。

3. 発明の詳細な説明

〔発明の技術分野〕

本発明は、データ処理装置におけるソースプログラムのコンパイル処理において、コンパイラの欠陥により無限ループにおちいつた状態を早期に検出するためのコンパイラ障害自動検出方式に関する。

〔技術的背景〕

一般に、コンパイル処理において、コンパイラ自身のバグで異常終了となる場合には、無限ループの場合と、それ以外の特定処理での障害の場合とがある。通常、コンパイラの異常終了ではメモリ内容ダンプの指定が行なわれていないので、後者の場合、メモリ内容ダンプの指定をとり直してもう1度処理を繰り返せばバグ位置をつきとめることが可能である。しかし、前者の無限ループに入つた場合には、たとえば30秒乃至60秒に散

定されているCPUの時間監視(CPU TIME OVER)にかかるまで、そのまま走行を続けた後、異常終了することになる。しかもこのとき、メモリ内容のダンプ指定が行なわれていないと、異常終了のメッセージが出力されるのみであり、また、たとえ異常終了時のダンプ指定が行なわれたとしても、ループのどの位置でCPU TIME OVERにかかるかは定まっていなため、異常終了の原因を知る確実な手掛りが得られず、障害回復にかなりの手間と時間がかかっていた。

#### 〔発明の目的および構成〕

本発明の目的は、コンパイラの欠陥により無限ループに入つた場合に、CPU時間監視によらずに異常を検出して、必要な情報を出力することを可能にする手段を提供することにある。そのため本発明は、コンパイラの処理が複数の処理段階(フェーズ)で構成されていて、ソースプログラムはこれらの処理段階を順次パスして行くことによりオブジェクトに変換される点と、各処理段階でのパス処理時間に極端な差がないことに着目して、

各処理段階のパス処理時間の間の相対的な比較によつて無限ループの検出を行なうものである。

そしてそれによる本発明の構成は、ソースプログラムを複数回のパスによりコンパイル処理するコンパイラをそなえたデータ処理装置において、上記コンパイラを用いたコンパイル処理の第1回のパスに要する処理時間と、第2回以降の各回ごとのパスに要する処理時間との間の比率について予測できる最大値を超えた所定の値が予め設定されているテーブル手段と、実際のソースプログラムについてコンパイル処理時に各回ごとのパスに要する処理時間を計測する手段と、第2回以降のパスごとに要した処理時間と第1回のパスに要した処理時間に上記テーブル手段に設定されている当該回の比率とを乗じた時間とを比較し、前者が後者よりも大きくなつたとき無限ループ状態と判定する手段とを有し、該判定手段が無限ループ状態と判定したパスにおいて異常終了処理を行なうことを特徴とするものである。

#### 〔発明の実施例〕

以下に、本発明の詳細を実施例にしたがつて説明する。

第1図は、コンパイラの主要な処理段階を示したもので、単語解析、構文解析、最適化、記憶域割当て、コード生成、アセンブルの各処理からなり、それぞれの段階で各種の記号表、リテラル表、中間テキストなどが生成される。

上記の各処理段階は、実際には10乃至30のパス処理に分けられている。そしてそれぞれのパスに要する処理時間は、ソース行数に大体比例することが言える。したがつて、最初のパスでの処理時間と他のパスでの処理時間とは、任意のソースに対して同じ比例関係を保つものと考えることができる。

そこで、最初のパスで処理時間を測定し、2番目以降のパスの処理時間が最初のパスの処理時間に予め定められた一定の係数を掛けて得られた時間を超えたならば、コンパイラループが生じているものと判定して異常終了する機構を設け、その

時点での処理情報を出力させる。

第2図は、本発明の1実施例装置の機能構成図である。図中、1はデータ処理装置、2はコンパイル処理部、3は処理情報記憶部、4はループ検出処理部、5は処理時間タイマ、6はループ判定係数テーブル、7は異常終了処理部、8は外部記憶装置、9はプリンタを示す。

コンパイル処理部2は、コンパイラの各処理段階のパス処理を順次実行する。処理情報記憶部3はコンパイル処理に必要な全ての情報を記憶する。コンパイル処理部2は、パス処理において、必要に応じて処理時間タイマ5のセットを要求するSTIMERマクロ、テストを要求するTTIMERマクロ、クリアを要求するCTIMERマクロを、それぞれ発行する。

ループ検出処理部4は、OSの1機能部として設けられ、STIMER、TTIMER、CTIMERの各マクロに回答して処理時間タイマ5を制御するとともに、処理時間タイマ5のタイムオーバーを監視し、タイムオーバーの場合に異常終了処理部7

を起動する。

ループ判定係数テーブル6は、2回目以降の各パス $P_2, P_3, \dots, P_i, P_N$ について、それぞれの処理時間に掛けるべき係数 $k_2, k_3, \dots, k_i, \dots, k_N$ を保持するテーブルである。

異常終了処理部7は、コンパイラループが検出されたとき、コンパイル処理部2の処理を終了させるとともに、処理情報記憶部3にある中間テキストをプリンタ9へ出力させる処理を行なう。

第3図は、上記した実施例装置の処理手順の説明図である。

コンパイル処理部2は第1回のパス $P_1$ の始めに $STIMER(T_A)$ のマクロを発行し、そしてパス $P_1$ の終りに $TTIMER(T_B)$ のマクロを発行する。ループ検出処理部4は、 $STIMER(T_A)$ に応じて処理時間タイマ5に $T_A$ をプリセットして、クロックのカウントダウン動作を行なわせ、そして $TTIMER(T_B)$ マクロに応じて、処理時間タイマ5の残り時間 $T_B$ を調べ、パス $P_1$ の処理時間 $T_1 = T_A - T_B$ を求めて記憶する。

次に、コンパイル処理部2は、パス $P_2$ の始めに $STIMER(k_2 \cdot T_1)$ マクロを発行し、終りに $CTIMER$ マクロを発行する。ループ検出処理部4は、 $STIMER(k_2 \cdot T_1)$ マクロに応じて、ループ判定係数テーブル6から、係数 $k_2$ を求め、これと先に記憶してあるパス $P_1$ の処理時間 $T_1$ とを掛け合わせて、その結果値を、処理時間タイマ5にプリセットする。そしてその後 $CTIMER$ マクロを受けとるまで処理時間タイマ5のタイムオーバーを監視する。

タイマの残り時間が零となつてタイムオーバー、すなわち $T_2 > k_2 \cdot T_1$ になつた場合には、ループと判定して異常終了処理部7に通知し、他方、タイマがタイムオーバーにならない間に $CTIMER$ マクロを受けとつた場合、すなわち正常終了の場合には、タイマをクリアする。この正常終了の場合には以上のような $P_i$ の処理を繰り返す。

異常終了処理部7は、任意のパス $P_i$ においてループ検出処理部4からループ検出を通知されると、出口ルーチンを呼び出して実行し、ループに

よるコンパイラ障害の発生をメッセージ出力するとともに、処理情報記憶部3の中の中間テキストをプリンタ出力し、異常終了処理(ABEND)を行なう。

#### [発明の効果]

以上のように、本発明によれば、最初のパス $P_1$ で無限ループに入つた場合を除いてコンパイラのループは確実に検出でき、しかもそのパス位置情報と中間テキストが得られるので、コンパイラのバグの検出が容易となり障害の回復処理に要する手間と時間の大幅な短縮が可能となる。

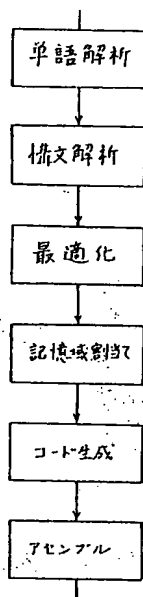
#### 4. 図面の簡単な説明

第1図はコンパイラの処理段階の説明図、第2図は本発明の1実施例装置の構成図、第3図は本発明実施例の処理手順のフロー図である。

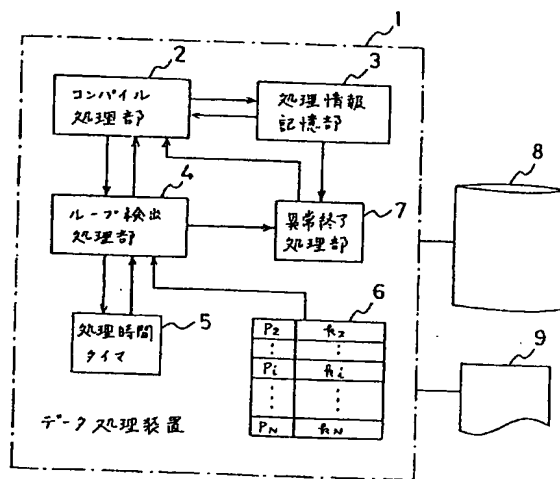
図中、1はデータ処理装置、2はコンパイル処理部、3は処理情報記憶部、4はループ検出処理部、5は処理時間タイマ、6はループ判定係数テーブル、7は異常終了処理部、8は外部記憶装置、

9はプリンタを示す。

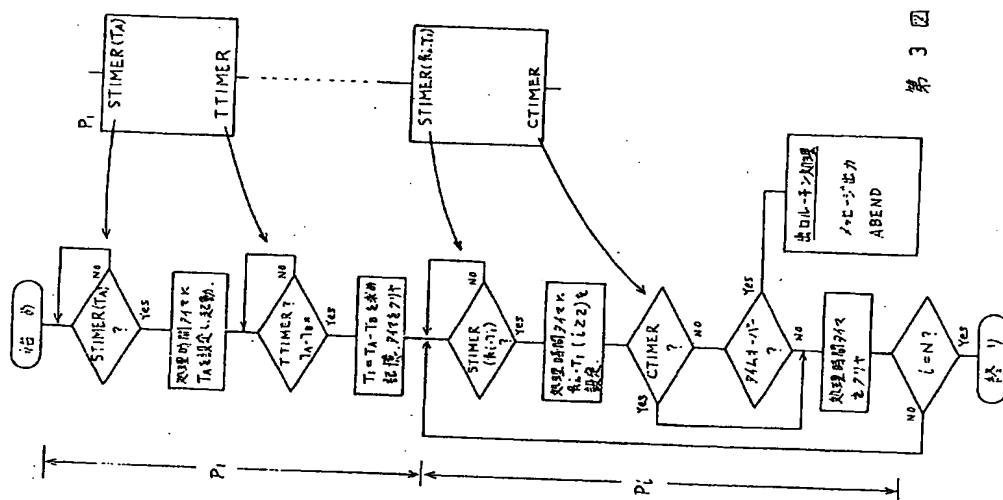
特許出願人 富士通株式会社  
代理人 井理士 長谷川 文 廣  
(外1名)



第 1 図



第 2 図



第 3 図

- TI - Method for computer-aided selection of software methods, for example software test methods
- AB - The method involves selection of boundary conditions by the user from those presented on a screen by a computer. Using the boundary conditions and a rule base describing the suitability of different software methods wrt. different boundary conditions, at least one software method suitable for the boundary conditions is selected and displayed by the computer.  
The method can be implemented using fuzzy logic. The boundary conditions can be described by technical boundary conditions, as aims or constraints, and can be associated with different degrees of significance. The degrees of significance can be associated with aims in pairs and can be subjected to consistency checking.

PN - EP0794489 A 19970910

AP - EP19970103549 19970304

OPD - 1996-03-07

PR - DE19961008889 19960307

PA - SIEMENS AG (DE)

IN - LIGGESMEYER PETER DR (DE)

EC - G06F11/00A3 ; G06F9/44G4

IC - G06F9/44 ; G06F11/00

CT - XP000672263 A [XP]; XP000672211 A [X]; XP000672374 A [A]

- CTNP - [XP] P. LIGGESMEYER: "Selecting engineering techniques using fuzzy logic based decision support" PROCEEDINGS IEEE SYMPOSIUM AND WORKSHOP ON ENGINEERING OF COMPUTER-BASED SYSTEMS, 11. - 15.März 1996, FRIEDRICHSHAFEN, DE, Seiten 427-434, XP000672263;
- [X] P. LIGGESMEYER: "Eine Methode zur Konstruktion von Prüfstrategien für Software" INFORMATIK FORSCHUNG UND ENTWICKLUNG, Bd. 9, Nr. 2, 1994, DE, Seiten 82-92, XP000672211;
- [A] P. LIGGESMEYER: "A set of complexity metrics for guiding the software test process" SOFTWARE QUALITY JOURNAL, Bd. 4, Nr. 4, Dezember 1995, UK, Seiten 257-273, XP000672374

© WPI / DERWENT

- TI - Computerised selection of software methods, e.g. software test methods, development methods, test metrics or development metrics - involves selection of boundary conditions by user from those presented by computer; selection of method using selected boundary conditions and associated rule base

- AB - EP-794489 The method involves selection of boundary conditions by the user from those presented on a screen by a computer. Using the boundary conditions and a rule base describing the suitability of different software methods wrt. different boundary conditions, at least one software method suitable for the boundary conditions is selected and displayed by the computer.
- The method can be implemented using fuzzy logic. The boundary conditions can be described by technical boundary conditions, as aims or constraints, and can be associated with different degrees of significance. The degrees of significance can be associated with aims in pairs and can be subjected to consistency checking.
- ADVANTAGE - Enables selection of optimal method from number of available methods.
- (Dwg.1/4)

PN - EP0794489 A2 19970910 DW199741 G06F9/44 Ger 017pp

OPD - 1996-03-07

PR - DE19961008889 19960307

PA - (SIEI ) SIEMENS AG

IN - LIGGESMEYER P

IC - G06F9/44 ; G06F11/00

**This Page Blank (uspto)**

# **Method for computer-aided selection of software methods, for example software test methods**

Patent Number: EP0794489  
 Publication date: 1997-09-10  
 Inventor(s): LIGGESMEYER PETER DR (DE)  
 Applicant(s): SIEMENS AG (DE)  
 Requested Patent: ☐ EP0794489, A3  
 Application Number: EP19970103549 19970304  
 Priority Number(s): DE19961008889 19960307  
 IPC Classification: G06F9/44 ; G06F11/00  
 EC Classification: G06F11/00A3, G06F9/44G4  
 Equivalents:

## **Abstract**

The method involves selection of boundary conditions by the user from those presented on a screen by a computer. Using the boundary conditions and a rule base describing the suitability of different software methods wrt. different boundary conditions, at least one software method suitable for the boundary conditions is selected and displayed by the computer. The method can be implemented using fuzzy logic. The boundary conditions can be described by technical boundary conditions, as aims or constraints, and can be associated with different degrees of significance. The degrees of significance can be associated with aims in pairs and can be subjected to consistency checking.

~~~~~  
 Data supplied from the esp@cenet database - 12

**This Page Blank (uspto)**



● EPODOC / EPO

TI - SYSTEM FOR DETECTING AUTOMATICALLY FAULT OF COMPILER  
PN - JP60118935 A 19850626  
AP - JP19830226321 19831130  
OPD - 1983-11-30  
PR - JP19830226321 19831130  
PA - FUJITSU LTD  
IN - IKEDA TAKAAKI  
IC - G06F9/44 ; G06F11/00

● PAJ / JPO

TI - SYSTEM FOR DETECTING AUTOMATICALLY FAULT OF COMPILER  
AB - PURPOSE: To output required information by comparing relative comparison between path processing times of each processing stage to detect an infinite loop, thereby detecting a fault independently of supervision of a CPU time.  
- CONSTITUTION: A compile processing section 2 executes sequentially a path processing of each processing stage of a compiler. A processing information storage section 3 stores all the information required for the compile processing. The processing section 2 issues an STIMER marco requesting the set of a processing time timer 5 as required, a T-TIMER macro requesting the test and a CTIMER macro requesting clear. A loop detecting processing section 4 is provided as one function section of an OS, the processing time timer 5 supervises timeover in response to each said macro. A loop discrimination coefficient table stores a coefficient multiplied with each path. An error end processing section 7 completes the processing when a compiler loop is detected and gives an output to a printer 9.  
PN - JP60118935 A 19850626  
AP - JP19830226321 19831130  
PA - FUJITSU KK  
IN - IKEDA TAKAAKI  
I - G06F9/44 ; G06F11/00

**This Page Blank (uspto)**